
# Congestion Control In The Internet

## Part 1: Theory

JY Le Boudec

2022

# Contents

1. What is the problem; congestion collapse
2. Efficiency versus Fairness
3. Definitions of fairness
4. Additive Increase Multiplicative Decrease (AIMD)
5. Slow start

## Textbook


Rate adaptation, Congestion Control and Fairness: A Tutorial

JEAN-YVES LE BOUDEC
*Ecole Polytechnique Fédérale de Lausanne (EPFL)*

# 1. Congestion Collapse

In October of '86, the Internet had the first of what became a series of 'congestion collapses'. During this period, the data throughput from LBL to UC Berkeley (sites separated by 400 yards and three IMP hops) dropped from 32 Kbps to 40 bps. Mike Karels[1] and I were fascinated by this sudden factor-of-thousand drop

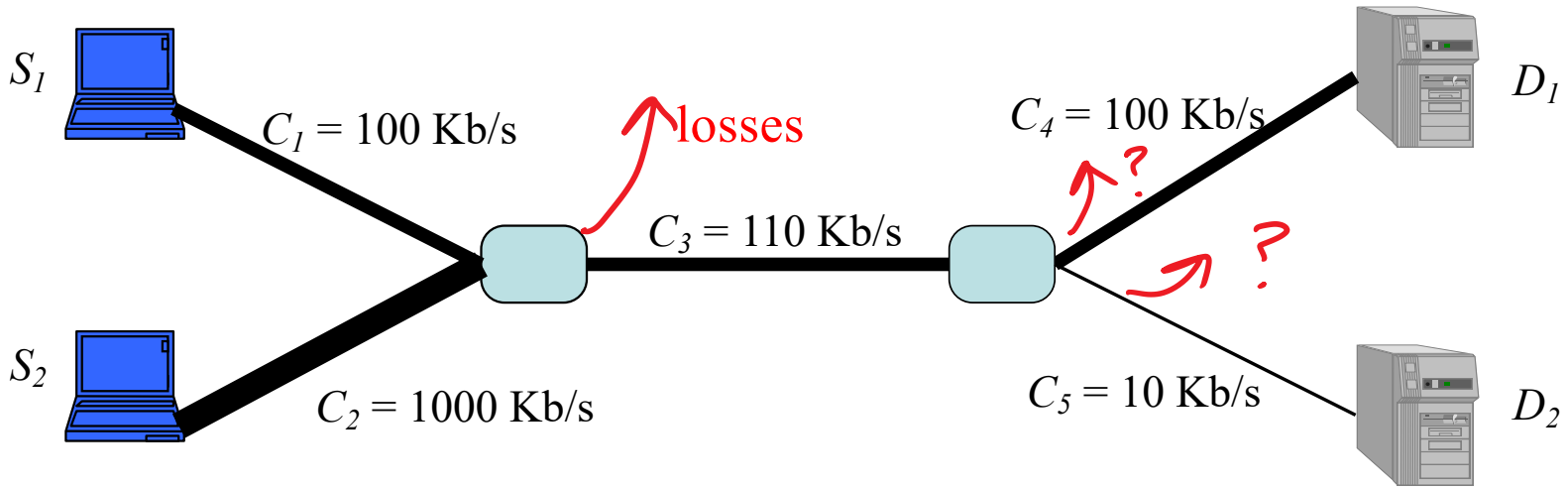
Jacobson, Van. "Congestion avoidance and control." *ACM SIGCOMM computer communication review*. Vol. 18. No. 4. ACM, 1988.

# How much will $S_1$ send to $D_1$ ? $S_2$ to $D_2$ ?

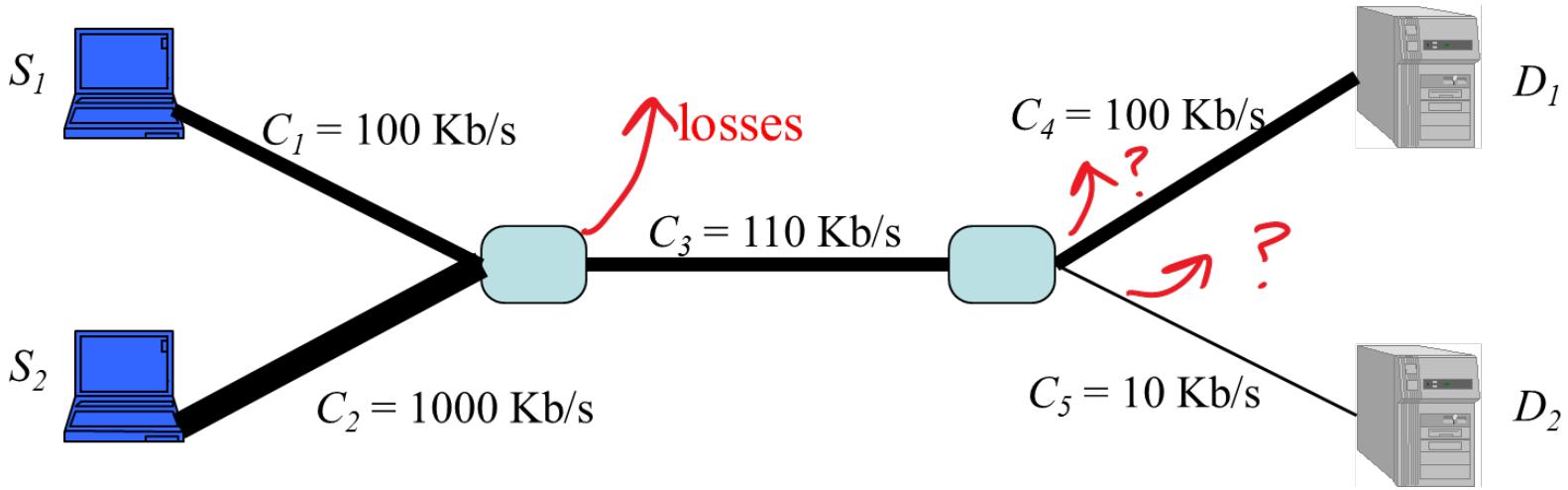
Network may lose some packets;

Assume greedy sources (i.e. send as much as they want)

Assume loss is proportional to submitted traffic and links can be fully utilized



# $S_1$ to $D_1$ rate is …




A. 10 kb/s
B. 50 kb/s
C. 100 kb/s
D. I don't know

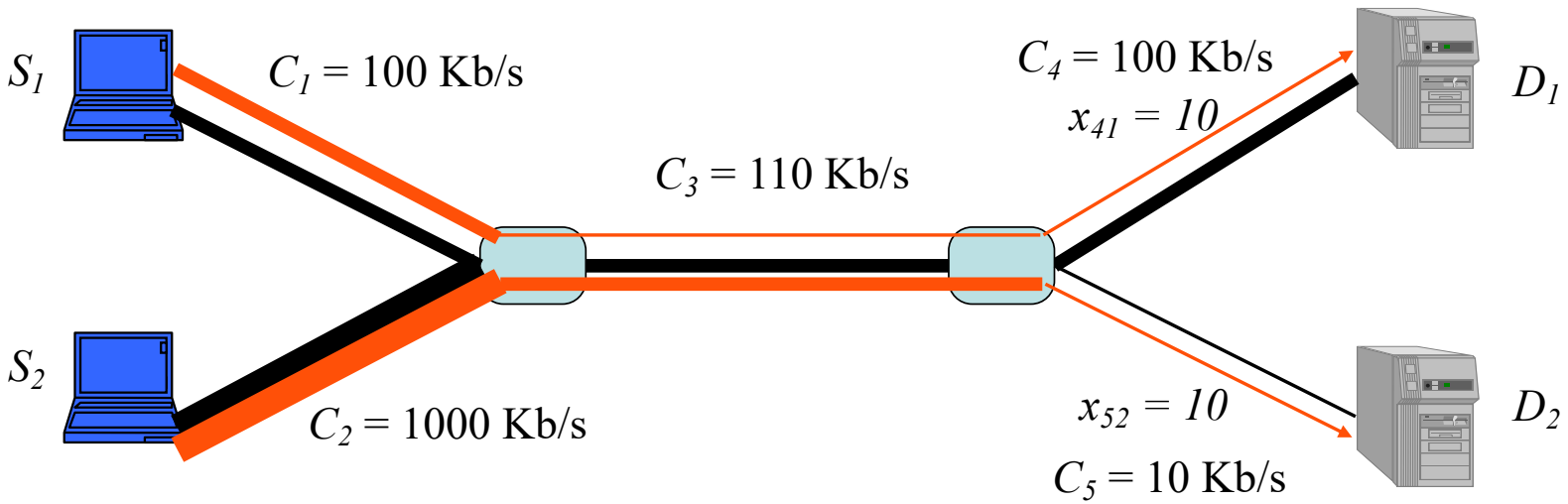
# Take home message
# Greedy Sources May Be Inefficient

A better allocation is:

$S_1$: 100 kb/s

$S_2$: 10 kb/s

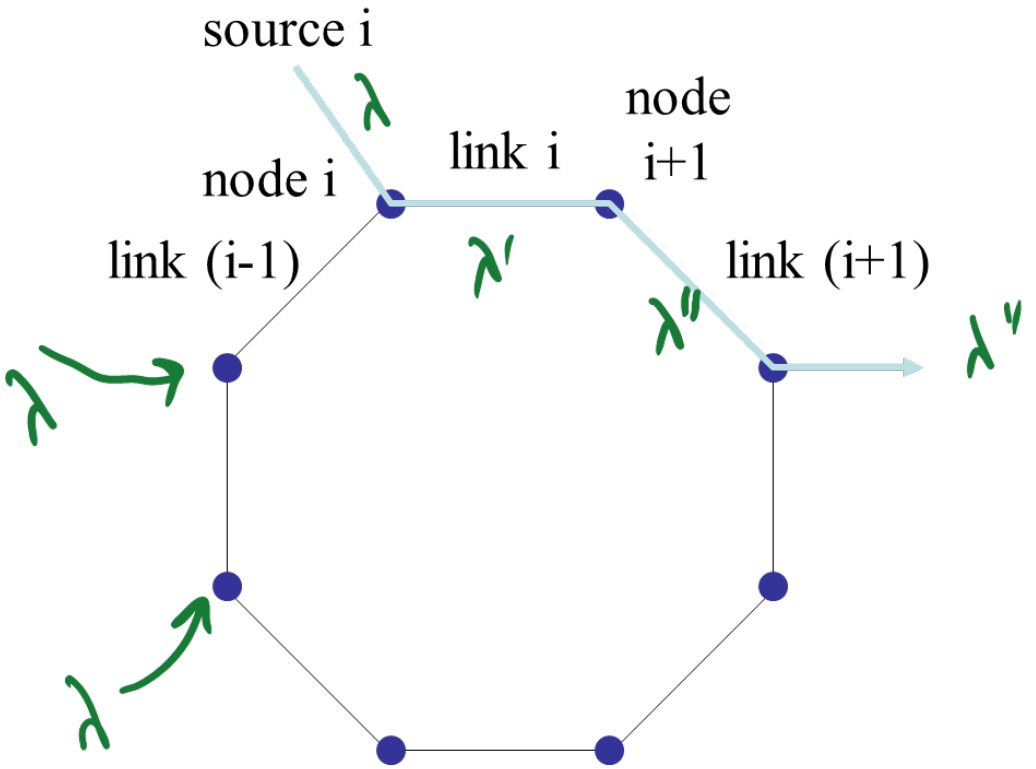
The problem was that $S_2$ sent too much (but did not know)



# How much can node $i$ send to its destination ?

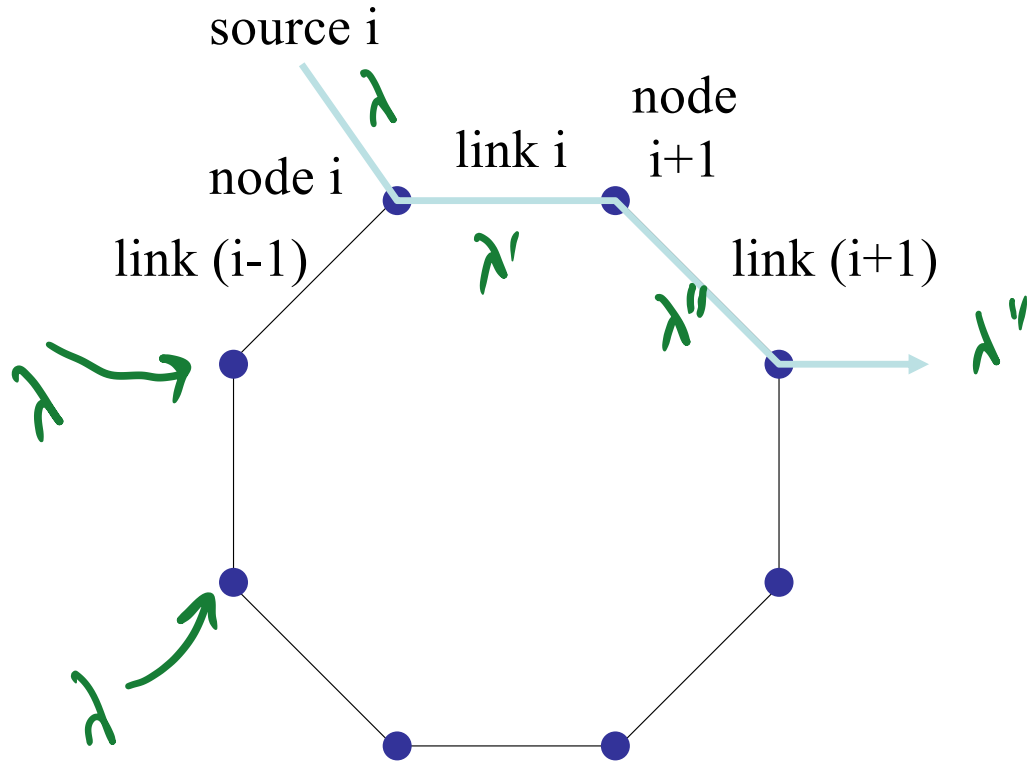
Source $i$ uses two links, all of same capacity $c$

At every node there is a source – all sources at same rate



# How much can node $i$ send to its destination ?

If $\lambda < \frac{c}{2}$ there is no loss (in this simple model) and $\lambda'' = \lambda$



# How much can node $i$ send to its destination ?

If $\lambda > \frac{c}{2}$ there is some loss (in this simple model)

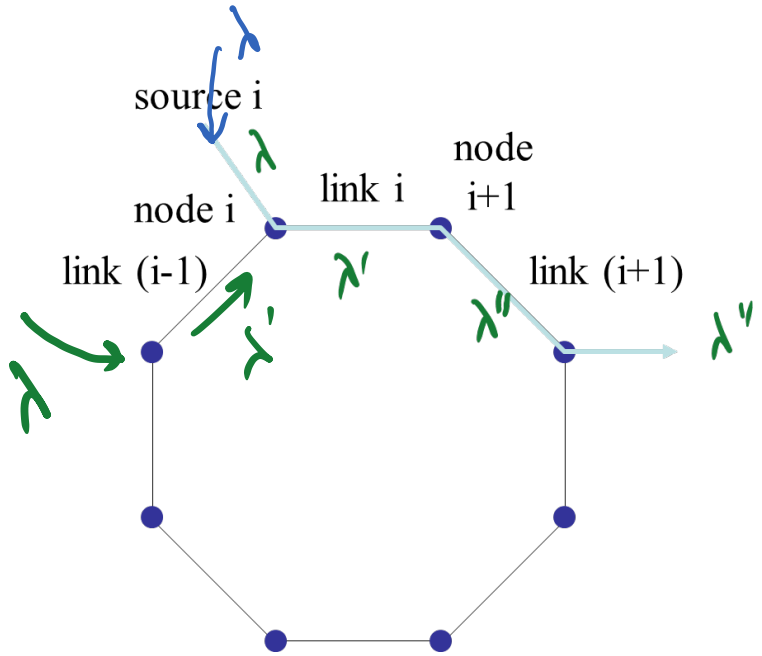
Ratio of accepted traffic at one node is $\frac{c}{\lambda+\lambda'}$ ← capacity / ← offered traffic

Therefore

$$\lambda' = \frac{c}{\lambda + \lambda'}\lambda \quad (1)$$

$$\lambda'' = \frac{c}{\lambda + \lambda'}\lambda' \quad (2)$$

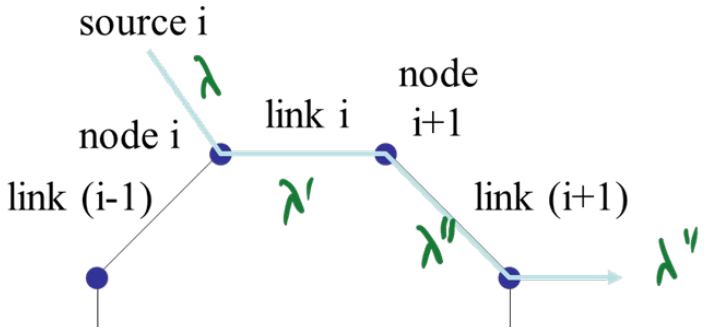
Solve for $\lambda'$,$\lambda''$ from (1),(2)

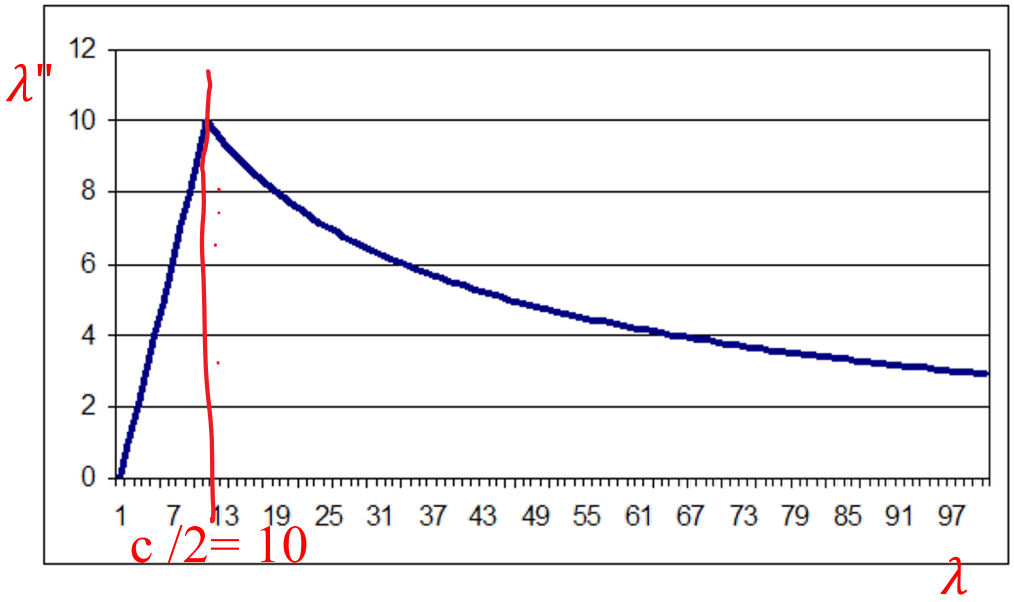


# How much can node $i$ send to its destination ?

We obtain, for $\lambda > \frac{c}{2}$ :

$$\lambda'' = c - \frac{\lambda}{2}\left(-1 + \sqrt{1 + \frac{4c}{\lambda}}\right)$$

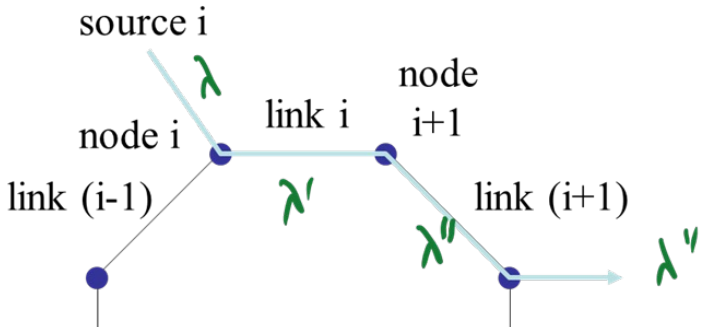


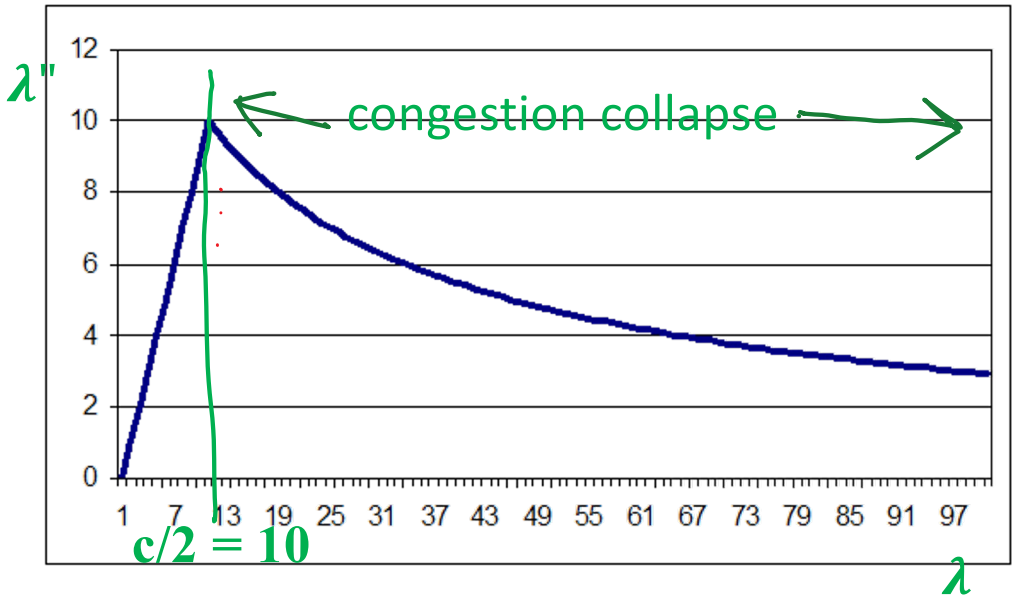



# For large offered traffic $\lambda$, the limit of useful work is 0

We obtain, for $\lambda > \frac{c}{2}$ :

$$\lambda'' = c - \frac{\lambda}{2}\left(-1 + \sqrt{1 + \frac{4c}{\lambda}}\right)$$




$$\sqrt{1+u} = 1 + \frac{1}{2}u - \frac{1}{8}u^2 + o(u^2)$$

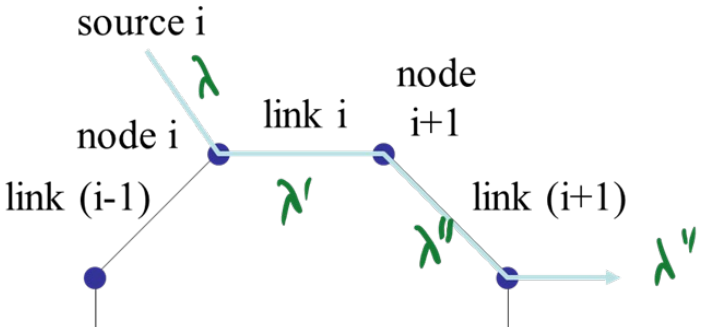
$$\lambda'' = \frac{c^2}{\lambda} + o(\frac{1}{\lambda})$$

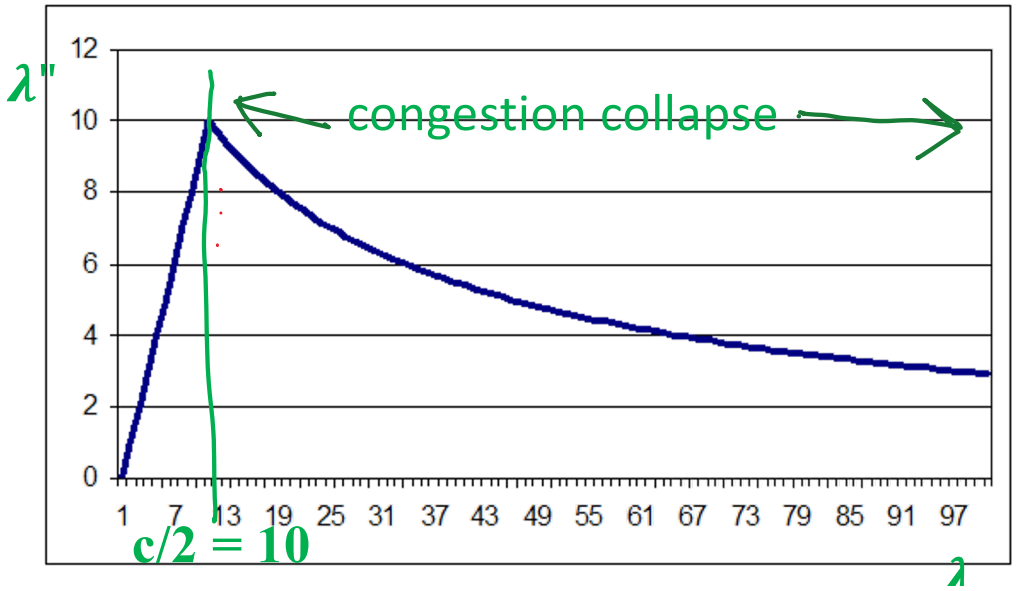
# Take-Home Message 2

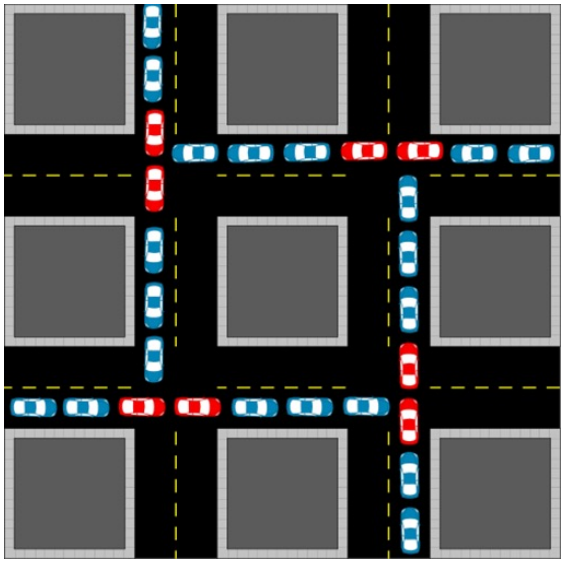
This *is* congestion collapse, i.e. : as the offered load increases, the total throughput decreases

Sources should limit their rates to adapt it to the network condition

Otherwise inefficiency or congestion collapse may occur



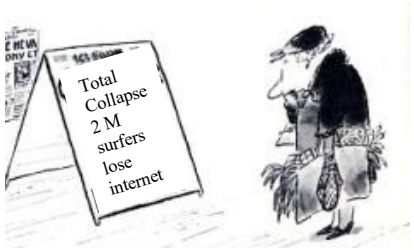



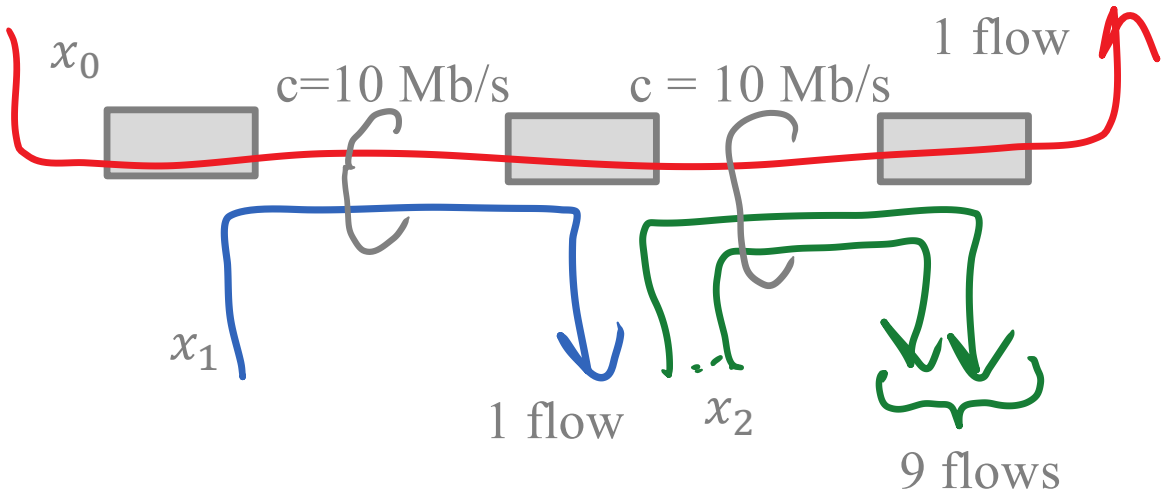
# 2. Efficiency vs Fairness

A network should be organized so as to avoid inefficiency

However, being maximally efficient may be a problem

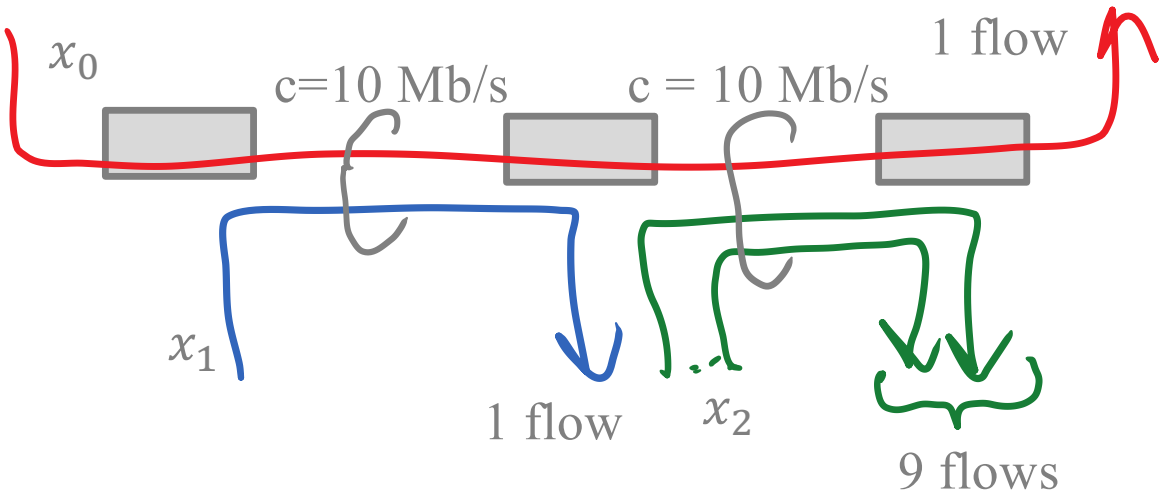
Example : what is the maximum total throughput in this network ?



# The maximum throughput is …

A. 5 Mb/s
B. 10 Mb/s
C. 20 Mb/s
D. None of the above
E. I don't know



## The value of $x_0$ when the maximum throughput is attained is …

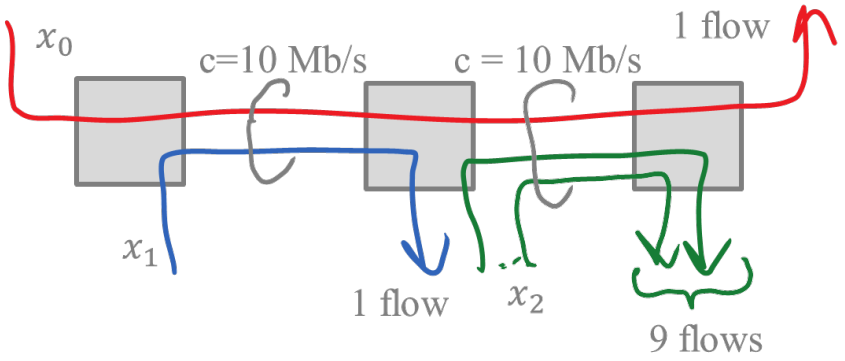
A. 1 Mb/s

B. $\frac{10}{9}$ Mb/s

C. 2 Mb/s

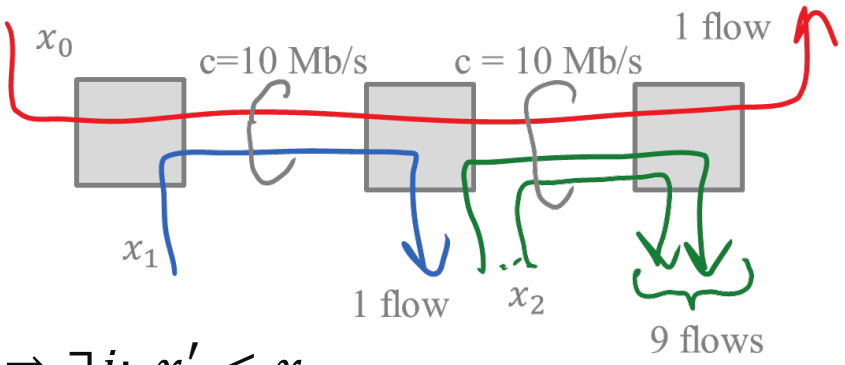
D. None of the above

E. I don't know



# Pareto Efficiency

A feasible allocation of rates $\vec{x}$ is called Pareto-efficient (synonym: Pareto-optimal) iff increasing one source must be at the expense of decreasing some other source




i.e. $\vec{x}$ is Pareto-efficient iff :

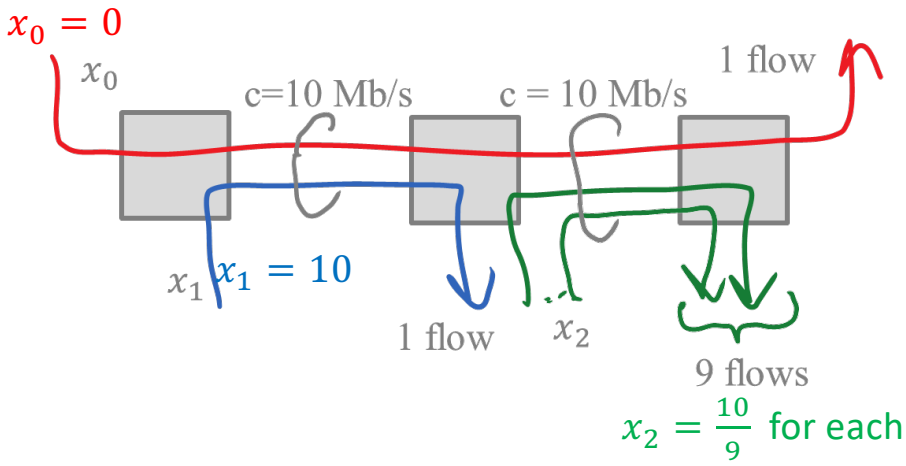
for any other feasible $\vec{x}', \exists i\colon\ x_i' > x_i \Rightarrow \exists j\colon\ x_j' < x_j$

$\Leftrightarrow$ every source has a bottleneck link (i.e. for every source $i$ there exists a link, used by $i$, which is saturated, i.e. the constraint of which is satisfied with equality).

An allocation $\vec{x}$ is not Pareto-efficient iff it can be improved unilaterally, i.e. there exists a feasible allocation $\vec{x}'$ such that $x_i' > x_i$ for some $i$ and $x_j' \geq x_j$ for all $j$.

# Example

Is the allocation
$x_0 = 0, x_1 = 10, x_2 = 10/9$
Pareto-efficient ?



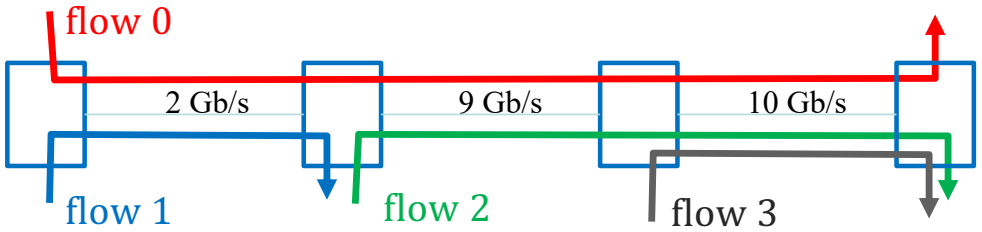

Link 1 is bottleneck for
sources 0 and 1
Link 2 is bottleneck for
source 0 and all sources 2

Every source has a bottleneck and cannot be increased unilaterally: The allocation is Pareto-efficient.

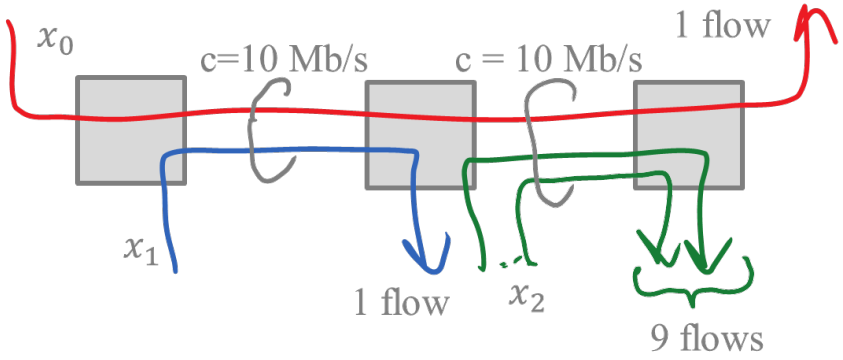
Observe that the throughput-maximizing allocation is always Pareto-efficient.

# Which allocations are Pareto-Efficient ?

A. $x_0 = 1, x_1 = 0.5, x_2 = 8, x_3 = 1$
B. $x_0 = 1, x_1 = 1, x_2 = 8, x_3 = 1$
C. $x_0 = 1, x_1 = 1, x_2 = 2, x_3 = 7$
D. A and B
E. A and C
F. B and C
G. All
H. None
I. I don't know



# Take Home Message



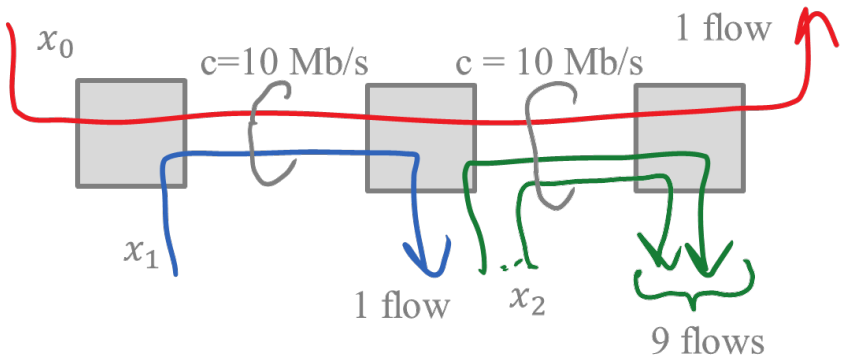

Maximal efficiency means Pareto efficiency.

Maximizing total throughput is Pareto efficient, but means shutting down flow 0; this is at the expense of fairness.

Are there Pareto efficient allocations that are fair ?

What is fairness ?

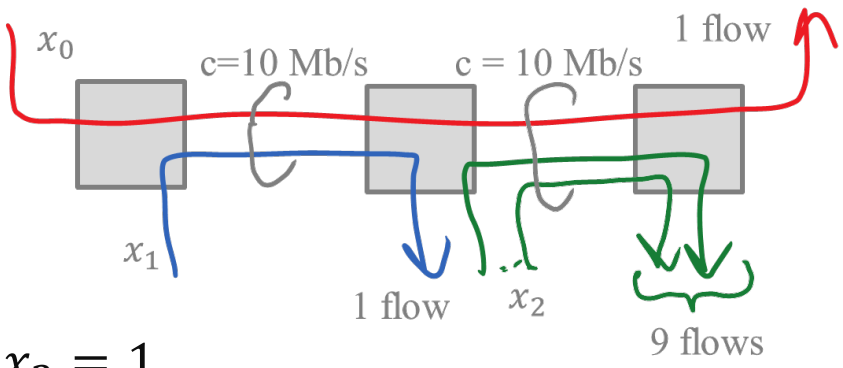
## 3. Let us try a first definition (Egalitarianism) : allocate as much as possible but same to all.
## In this example, what is the allocation according to this definition ?




A. $x_0 = x_1 = x_2 = 0.5\ Mb/s$
B. $x_0 = x_1 = x_2 = 1\ Mb/s$
C. $x_0 = x_1 = x_2 = \frac{10}{9}\ Mb/s$
D. None of the above
E. I don't know

# Egalitarianism is not Pareto- efficient

Egalitarianism gives $x = 1$ Mb/s to all this is stupid, we could give more to $x_1$ without hurting anyone




A better allocation is:

$$x_0 = 1, \qquad x_1 = 9, \qquad x_2 = 1$$

It is Pareto-efficient (every resource has a bottleneck) and is “fair”, since it gives to every one at least as much as egalitarianism.

This is the max-min fair allocation for this example.

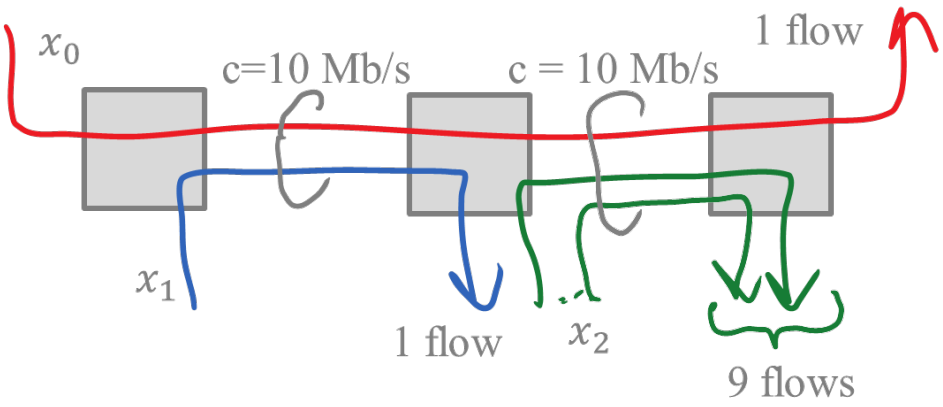
# Max-Min fairness

We say that a feasible allocation $\vec{x}$ is *max-min fair* iff for any other feasible $\vec{x}'$, $(\exists i\colon x_i' > x_i \Rightarrow \exists j\colon x_j' < x_j$ and $x_j \leq x_i)$

i.e. For every source $i$, increasing its rate must force the rate of some other (not richer) source $j$ to decrease

# Which allocations are max-min fair ?

| A | B |
|---|---|
| $x_0 = 0$ Mb/s | $x_0 = 1$ Mb/s |
| $x_1 = 10$ Mb/s | $x_1 = 9$ Mb/s |
| $x_2 = \frac{10}{9}$ Mb/s | $x_2 = 1$ Mb/s |




A. A
B. B
C. A and B
D. None
E. I don't know

# The Maths of Max-Min Fairness

Given a set of constraints for the rates

If it exists, the max-min fair allocation is *unique*

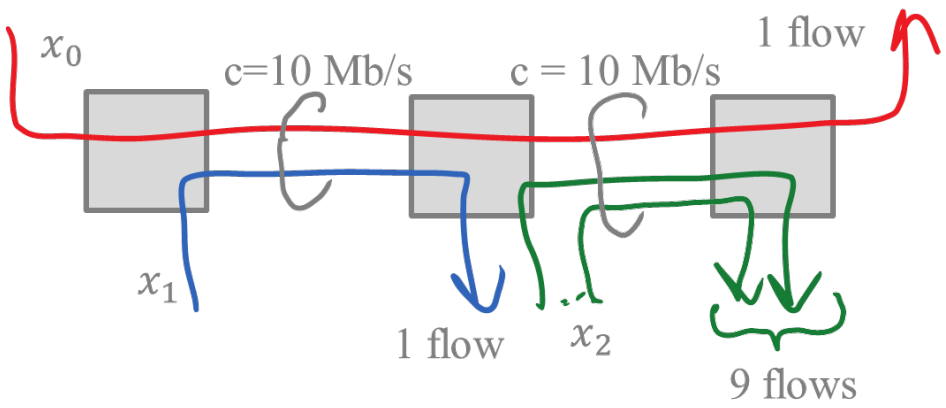
There *exists* one max-min fair allocation if the set of feasible rates is convex (this is the case for networks, as we have linear constraints)

The max-min fair allocation is Pareto-efficient (converse is not true)

For a set of feasible rates as in our case (the sum of the rates on every link is upper bounded), the (unique) max min fair allocation is obtained by *water-filling*

1 mark all sources as non frozen
2 **Do**
3 increase the rate of all non frozen sources to the largest possible common value
4 mark sources that use a saturated link as frozen
5 **Until** all sources are frozen

# Water-Filling Example




Step 1:

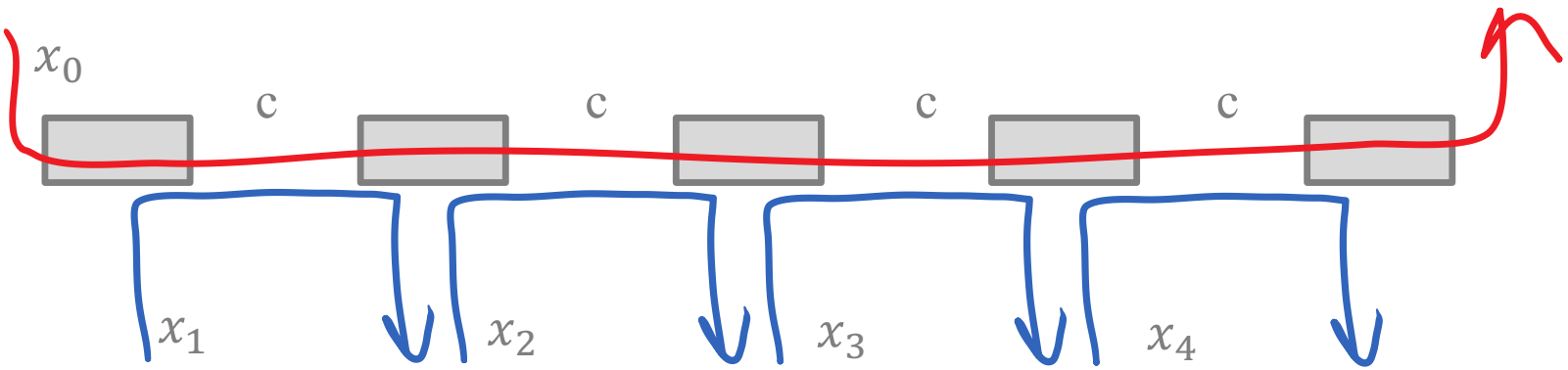
- maximize $t$ such that $x_0 = x_1 = x_2 = t$ and all constraints are satisfied; we find $t = 1$, hence $x_0 = x_1 = x_2 = 1$ ;
- link 2 is saturated, is used by sources 0 and 2 ⇒ mark sources 0 and 2 as frozen

Step 2 :

- maximize $t$ such that $x_1 = t$ , with $x_0 = 1, x_2 = 1$ and all constraints are satisfied; we find $t = 9$, hence $x_0 = x_2 = 1$ and $x_1 = 9$
- link 1 is saturated, is used by sources 0 and 1 ⇒ mark source 1 as frozen; all sources are frozen, STOP.

The max-min fair allocation is $x_0 = x_2 = 1$ and $x_1 = 9$

# What is the max-min fair allocation ?




A. $x_i = \frac{c}{2} \ \forall i$

B. $x_0 = \frac{c}{3}, \ x_i = \frac{2c}{3} \ \forall i \neq 0$

C. $x_0 = \frac{c}{4}, \ x_i = \frac{3c}{4} \ \forall i \neq 0$

D. $x_0 = \frac{c}{5}, \ x_i = \frac{4c}{5} \ \forall i \neq 0$

E. None of the above

F. I don't know

# Definition of Proportional Fairness

We say that a feasible allocation $\vec{x}$ is *proportionally fair* iff $\vec{x} > 0$ and for any other feasible $\vec{x}'$, $\sum_i \frac{x_i' - x_i}{x_i} \leq 0$

In other words: an allocation is proportionnally fair if for any other allocation, the total *rate of change* $\sum_i \frac{\Delta x_i}{x_i}$ is $\leq 0$

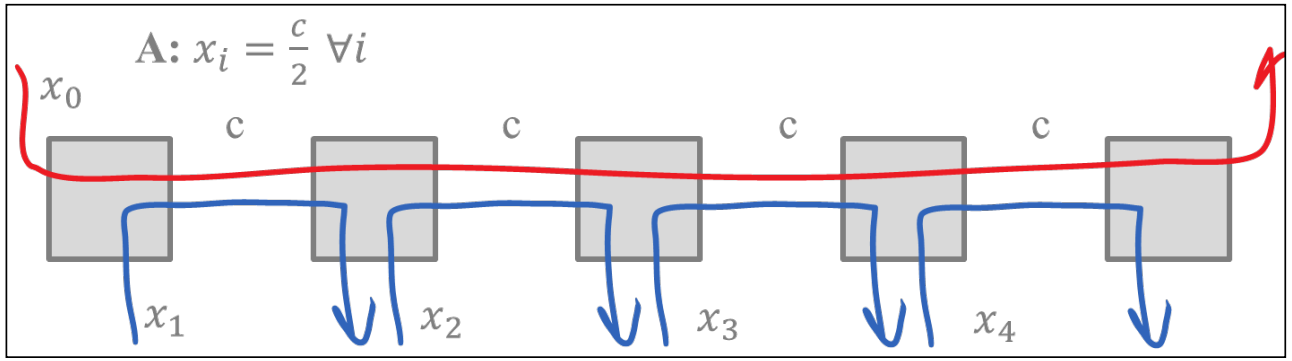
An allocation $\vec{x} > 0$ is *not proportionally fair* iff there is some other allocation $\vec{x}'$ such that $\sum_i \frac{x_i' - x_i}{x_i} > 0$

Two effects

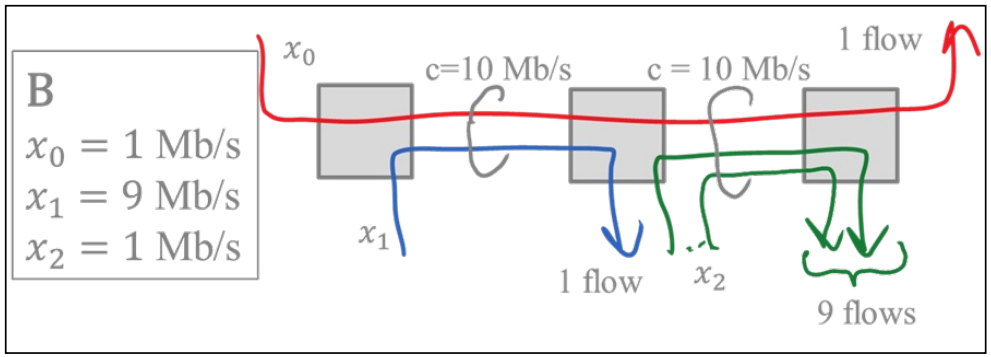
Relative shares matter, not absolute

Sum of all rates of changes

# Which allocations are proportionally fair ?




A. A
B. B
C. A and B
D. None
E. I don't know



# The Maths of Proportional Fairness

1. A proportionally fair allocation is Pareto-efficient.

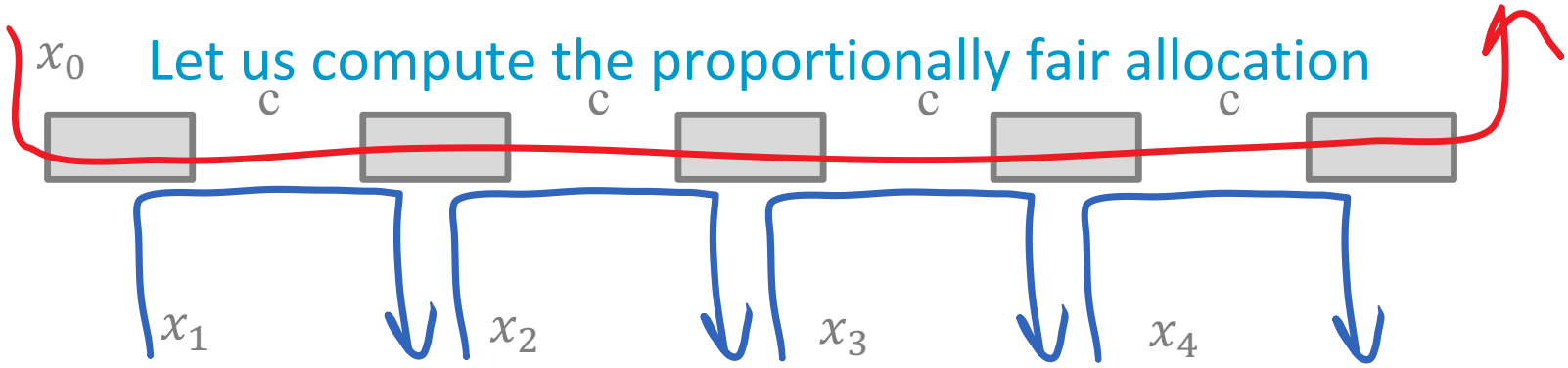
2. Given a set of constraints for the rates that is convex:
   The proportionally fair allocation *exists* and is *unique*
   It is obtained by maximizing

$$J(\vec{x}) := \sum_i \log x_i$$

   over all feasible allocations

   Intuitive explanation:

$$dJ\ (\vec{x}) = \sum_i \frac{dx_i}{x_i}$$



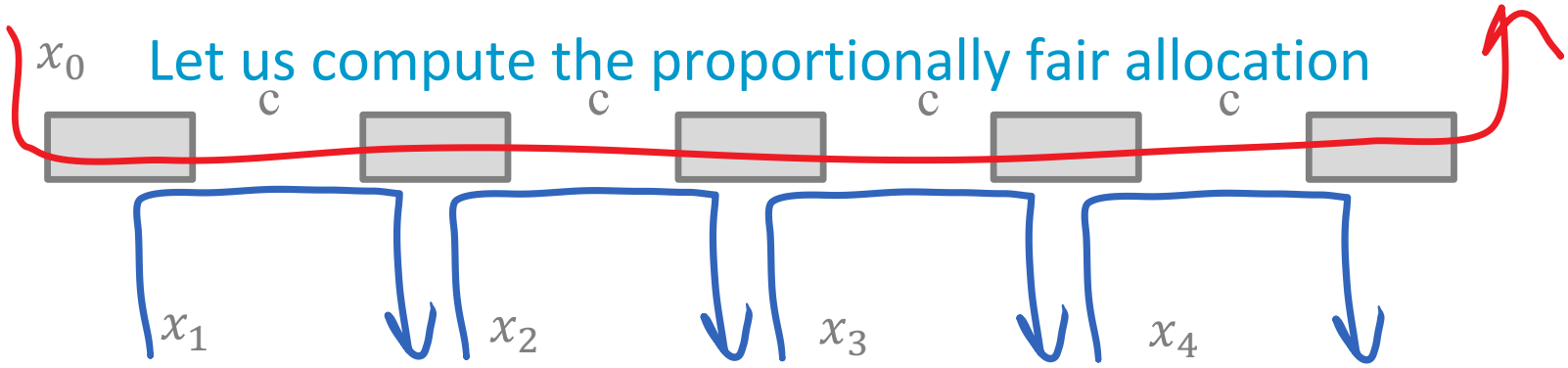

We have to solve the optimization problem:

$$\max U = \log x_0 + \log x_1 + \log x_2 + \log x_3 + \log x_4$$

subject to

$$x_0 + x_1 \leq c$$
$$x_0 + x_2 \leq c$$
$$x_0 + x_3 \leq c$$
$$x_0 + x_4 \leq c$$

We can use convex programming, but here we can also do a direct solution




We have to solve the optimization problem:

$$\max U = \log x_0 + \log x_1 + \log x_2 + \log x_3 + \log x_4$$

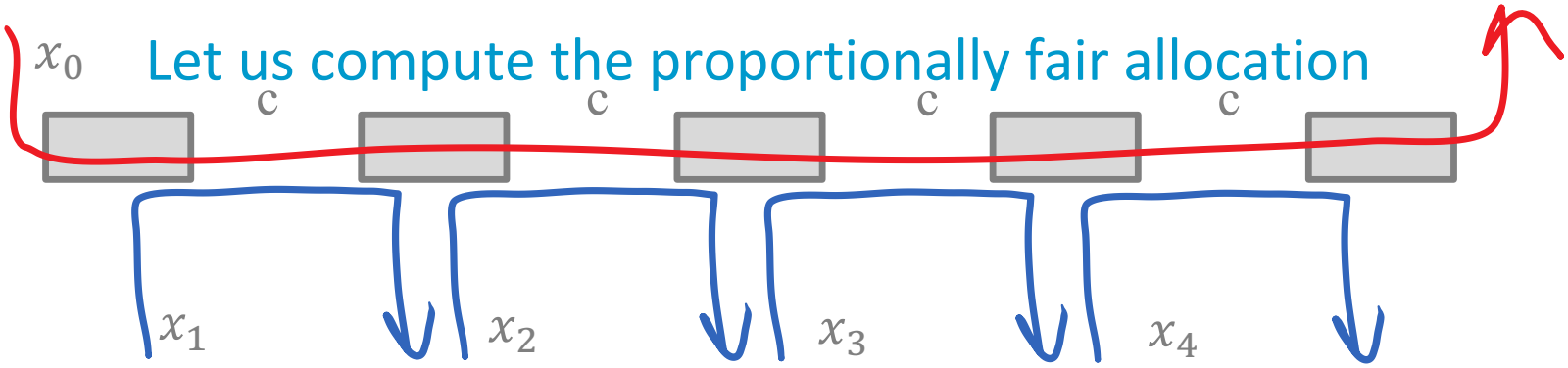
subject to

$$x_0 + x_1 \leq c$$
$$x_0 + x_2 \leq c$$
$$x_0 + x_3 \leq c$$
$$x_0 + x_4 \leq c$$

We must have equality everywhere otherwise we can increase $x_i$ $(i \neq 0)$ and increase *U*

Therefore we must have $x_1 = x_2 = x_3 = x_4 = c - x^*$
and $x_0 = x^*$ for some $x^*$

# Let us compute the proportionally fair allocation




We have to solve the optimization problem:

$$\max U = \log x^* + 4\log(c - x^*)$$
$$\text{subject to } 0 < x^* < c$$

This is a 1d problem, can be solved by computing the derivative

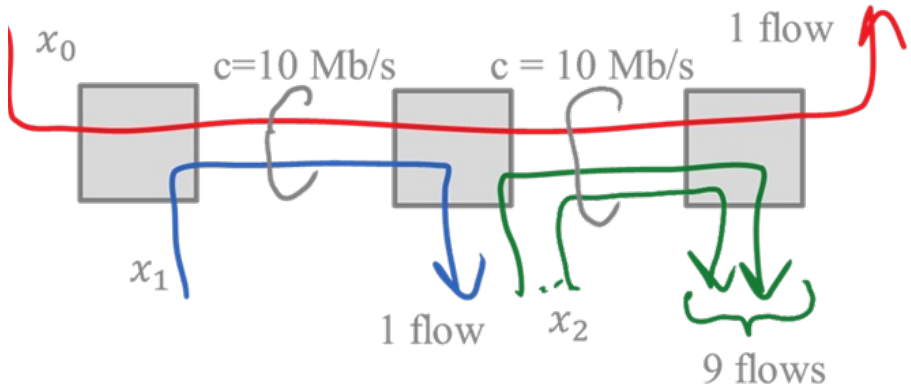
We find $\frac{dU}{dx^*} = \frac{1}{x^*} - \frac{4}{c-x^*}$

There is a maximum for $x^* = \frac{c-x^*}{4}$ i.e. $x^* = \frac{c}{5}$

The proportionally fair allocation is

$$x_0 = \frac{c}{5}, \qquad x_1 = x_2 = x_3 = x_4 = \frac{4c}{5}$$

# Which one is the proportionally fair allocation ? (in Mb/s) (only one answer)




A. $x_0 = 1, x_1 = 9, x_2 = 1$
B. $x_0 = 0.909, x_1 = 9, x_2 = 1.010$
C. $x_0 = 1.009, x_1 = 8.991, x_2 = 0.999$
D. $x_0 = 0.909, x_1 = 9.091, x_2 = 1.010$
E. I don't know

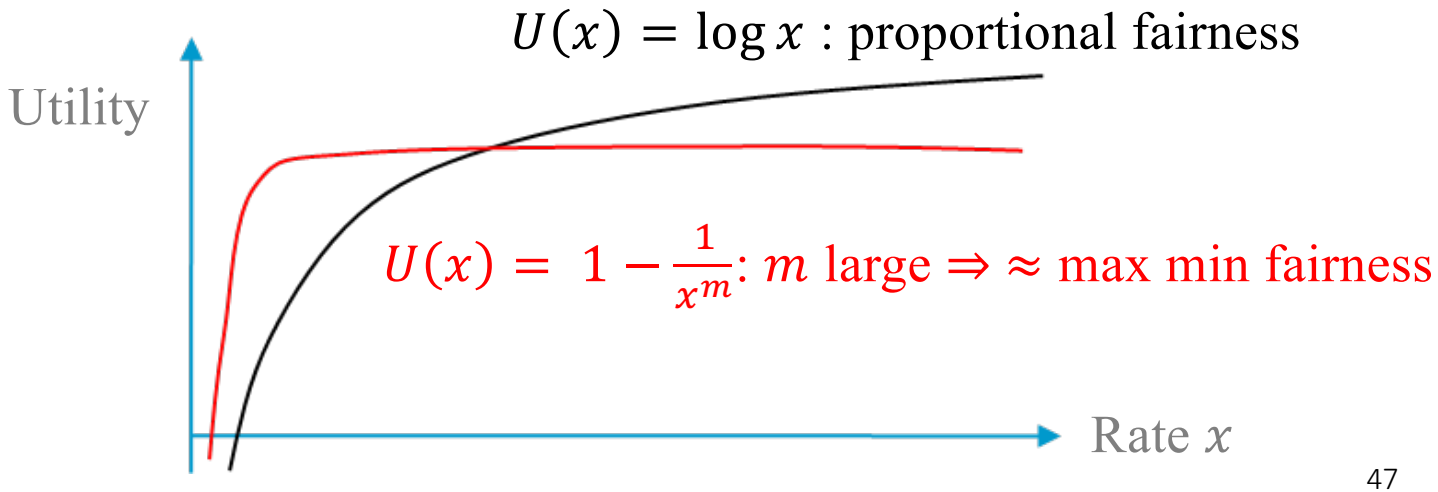
# Utility Fairness

One can *interpret* proportional fairness as the allocation that maximizes a global utility $\sum_i U_i(x_i)$ with $U_i(x_i) = \log x_i$.

If we take some other utility function we have what is called a utility fairness

It can be shown that max-min fairness is the limit of utility fairness when the utility function converges to a step function

but max-min fairness cannot be expressed exactly as a utility fairness



# Take Home Message

Sources should adapt their rate to the state of the network in order to avoid inefficiencies and congestion collapse

This is called "congestion control"

A rate adaptation mechanism should target some form of fairness

E.g. max-min fairness or proportional fairness

# 4. Additive Increase Multiplicative Decrease

How can congestion control be implemented ?

Explicit (rate based): tell every host how fast it can send

MPLS networks (smart grid)

Cellular networks

Hop by hop = backpressure: STOP/GO signals sent upstream

Gigabit LAN switches

Fair Queuing per Flow: One queue per flow / per user, served round robin

Cellular networks, industrial networks, in-vehicle networks

End-to-end: hosts taste the water and increase or decrease their sending rate using a host congestion control algorithm

The solution in the Internet

# Additive Increase Multiplicative Decrease (AIMD)

The first congestion control algorithm deployed in the Internet and before that, in Decnet (the “Decbit”).

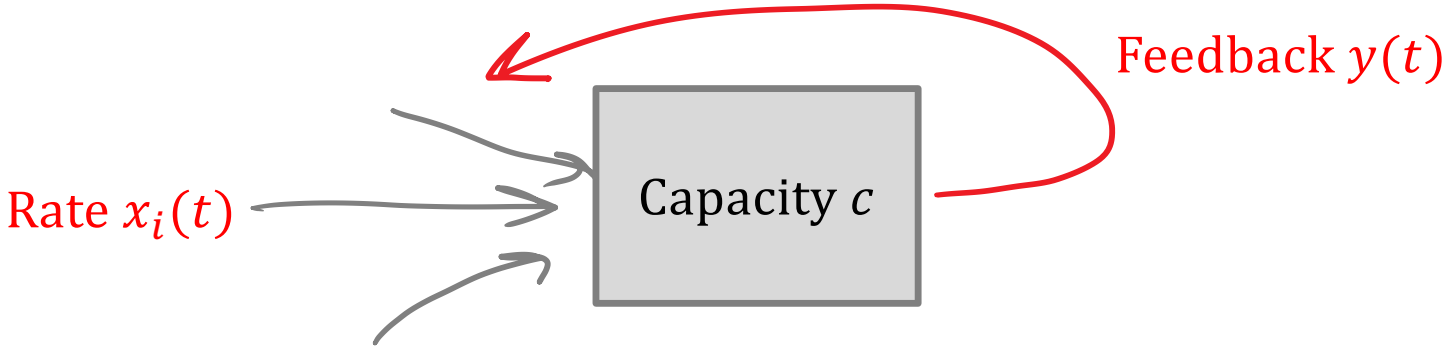
Still widely deployed today

We have designed a scheme that allows a network to operate at its knee. As shown in Figure 3, the scheme uses one bit called the **congestion avoidance bit** in the network layer header of the packet for feedback from the subnet to the users. A source clears the congestion avoidance bit as the packet enters the subnet. All routers in the subnet monitor their load and if they detect that they are operating above the knee, they set the congestion avoidance bit in the packets belonging to users causing overload. Routers operating below the knee pass the bit as received. When the packet is received at the destination the network layer passes the bit to the destination transport, which takes action based on the bits.

Raj Jain

Raj Jain, K.K. Ramakrishnan, and Dah-Ming Chiu. Congestion avoidance in computer networks with a connectionless network layer. Technical Report DEC-TR-506, Digital Equipment Corporation, August 1987.

# A Simple Network Model



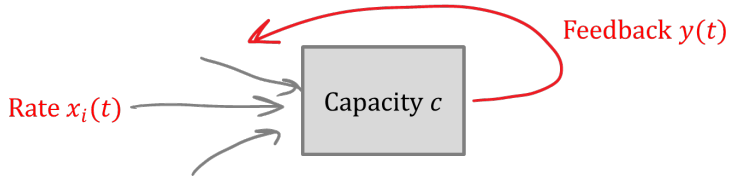

Network sends a one-bit feedback :

$y(t) = 0$ if $\sum_i x_i(t) \leq c$ , $y(t) = 1$ if $\sum_i x_i(t) > c$

Sources reduce rate $x_i(t+1)$ if $y(t) = 1$, increase otherwise

Question: what form of increase/decrease laws should one pick?

# Linear Laws




We consider linear laws

if $y(t) = 1$ then $x_i(t+1) = u_1\, x_i(t) + v_1$

if $y(t) = 0$ then $x_i(t+1) = u_0\, xi(t) + v_0$

We want to decrease when $y(t) = 1$, so

$u_1 \leq 1$ and $v_1 \leq 0$ and at least one inequality must be strict

Multiplicative decrease factor

Additive decrease term

We want to increase when $y(t) = 0$, so

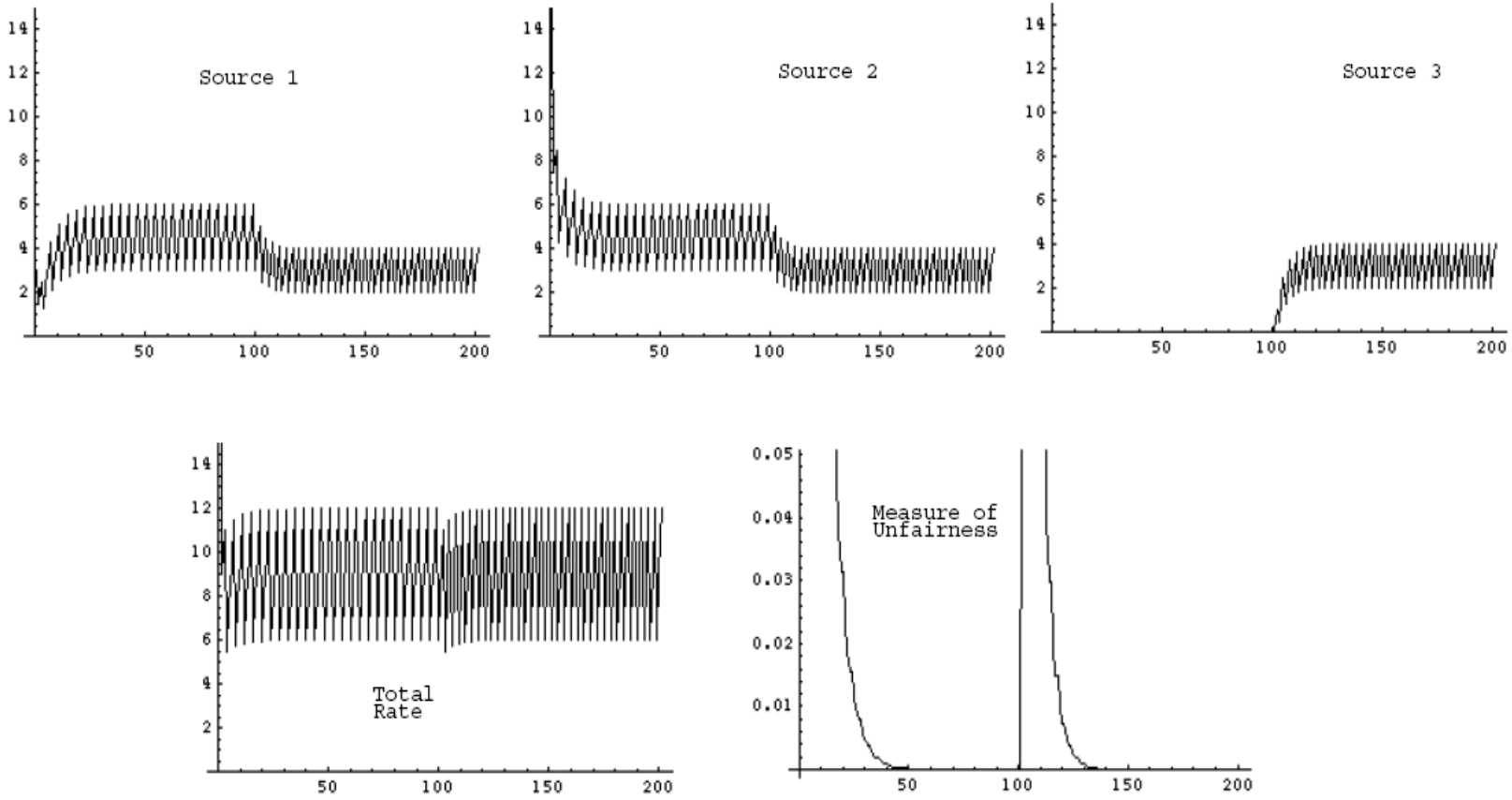
$u_0 \geq 1$ and $v_0 \geq 0$ and at least one inequality must be strict

Multiplicative increase factor

Additive increase term

# Example

$u_1 = 0.5, v_1 = 0$ (multiplicative decrease)
$u_0 = 1, v_0 = 1$ (Mb/s) (additive increase)



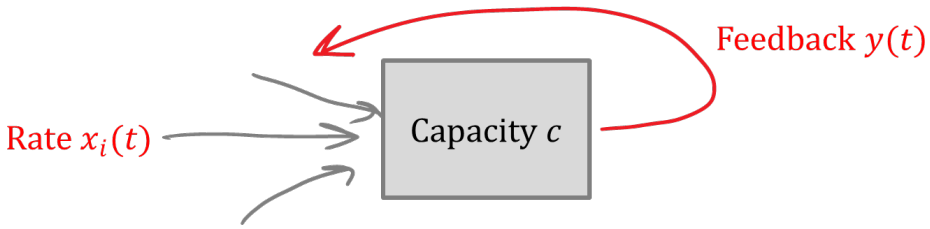
# Analysis of Linear Control Schemes

We want to achieve efficiency and fairness

We could target either max-min fair or proportionally fair allocations

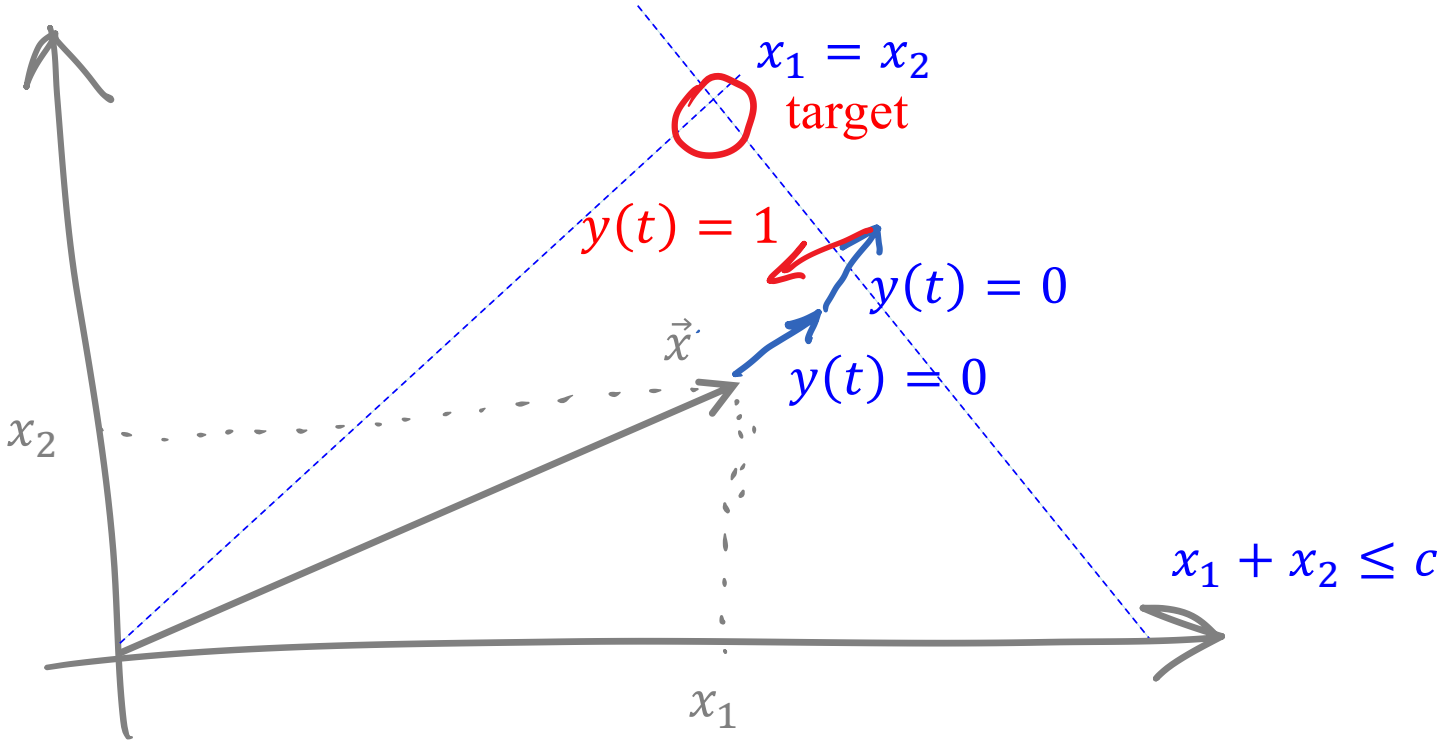
Here they are the same

We will now analyze the impact of each of the four coefficients $u_0, u_1, v_0$ and $v_1$.

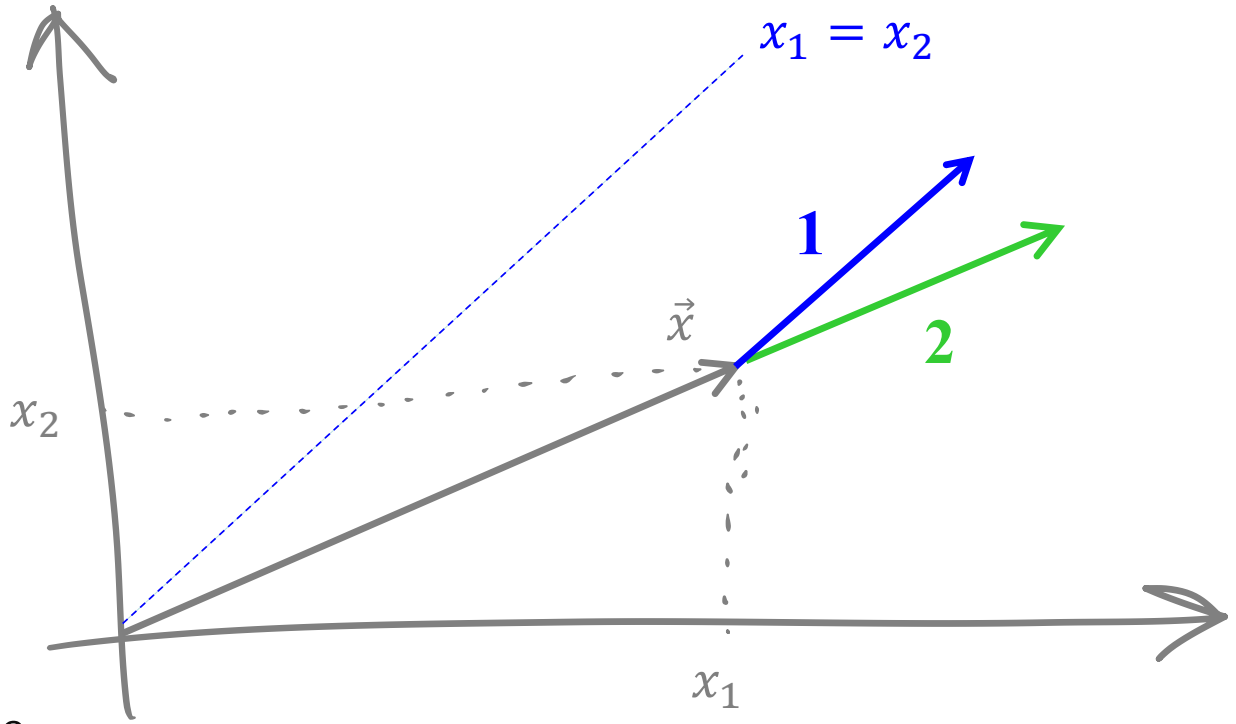



- We consider linear laws
  - if $y(t) = 1$ then $x_i(t+1) = u_1\, x_i(t) + v_1$
  - if $y(t) = 0$ then $x_i(t+1) = u_0\, xi(t) + v_0$
- We want to decrease when $y(t) = 1$, so
  - $u_1 \leq 1$ and $v_1 \leq 0$ and at least one inequality must be strict
  - $u_1$: Multiplicative decrease factor
  - $v_1$: Additive decrease term
- We want to increase when $y(t) = 0$, so
  - $u_0 \geq 1$ and $v_0 \geq 0$ and at least one inequality must be strict
  - $u_0$: Multiplicative increase factor
  - $v_0$: Additive increase term

# Zoom on 2 Sources



# Zoom on 2 sources ; say what is true




A. 1 = additive increase,
   2 = multiplicative increase
B. 1 = multiplicative increase
   2 = additive increase,
C. None of the above
D. I don't know

# Why AIMD

Among the linear controls, only additive increase – multiplicative decrease (AIMD) tends to bring the allocation towards fairness and efficiency.

This is what was implemented in the Internet after the first congestion collapses.

# In a more complex network setting, does AIMD distribute rates according to max-min fairness or proportional fairness ?

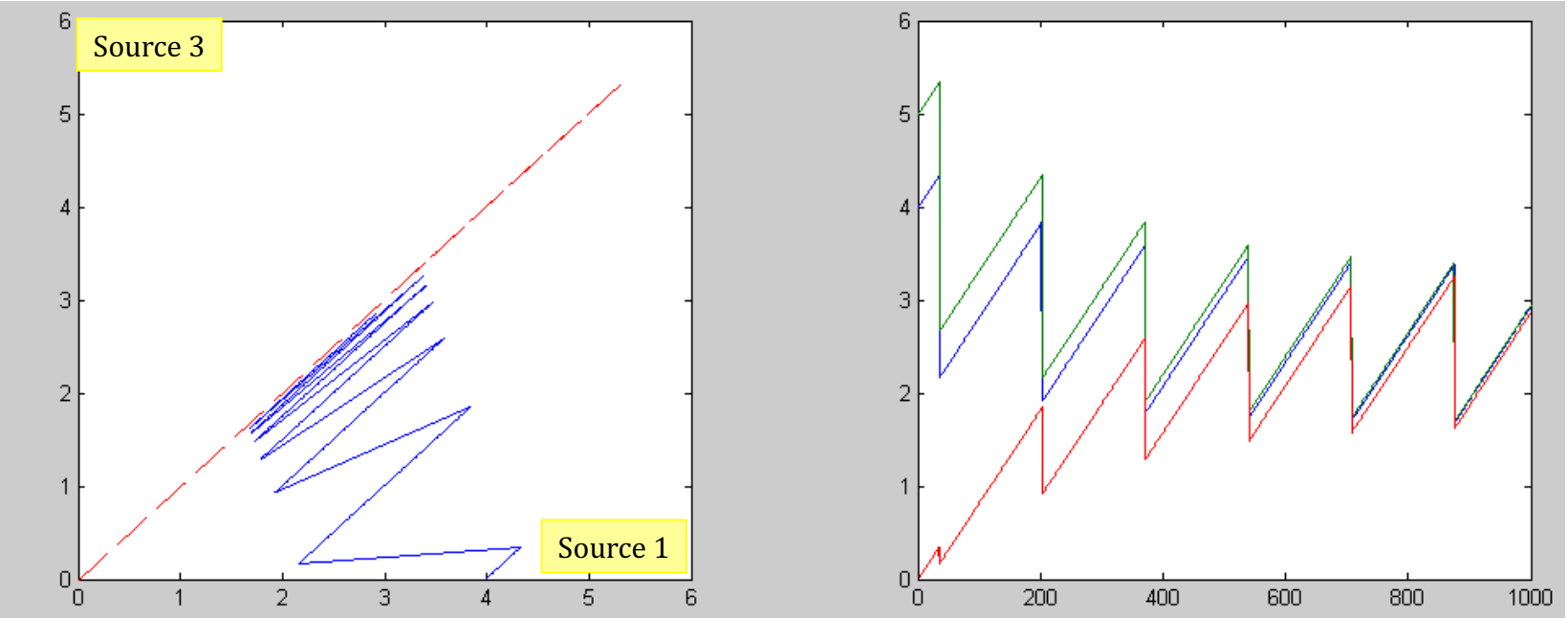
A. Max-min
B. Proportional
C. None of the above
D. I don’t know

# 5. Slow Start

AIMD convergence can be accelerated when initial conditions are very different

Slow start is an additional method, added to AIMD

Used at beginning of connection and at losses detected by timeout



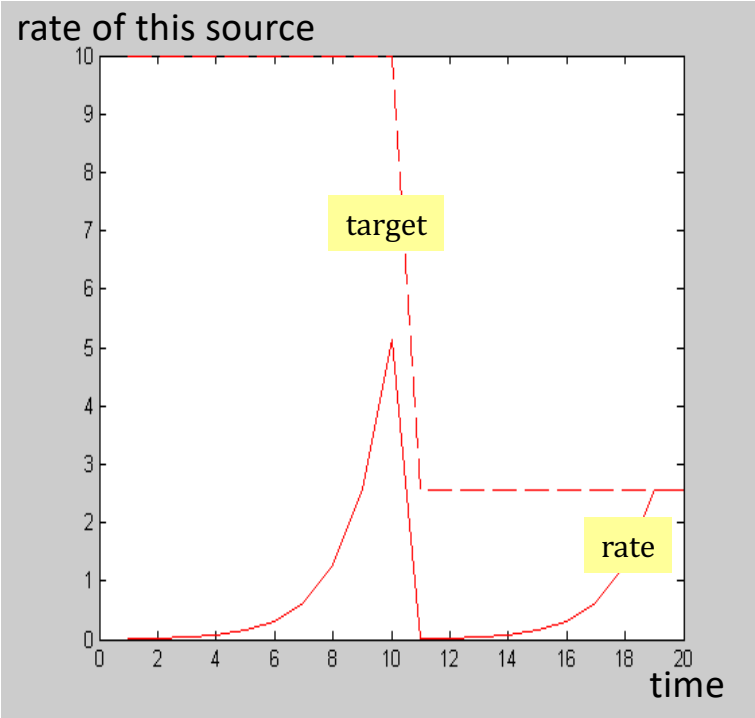
# Slow Start

Increase the rate multiplicatively

(by $w_0$, e.g. $w_0 = 2$) until a target rate is reached or negative feedback is received

Apply multiplicative decrease (by $u_1$, e.g. $u_1 = 0.5$) to target rate if negative feedback is received.

Exit slow start when target rate is reached

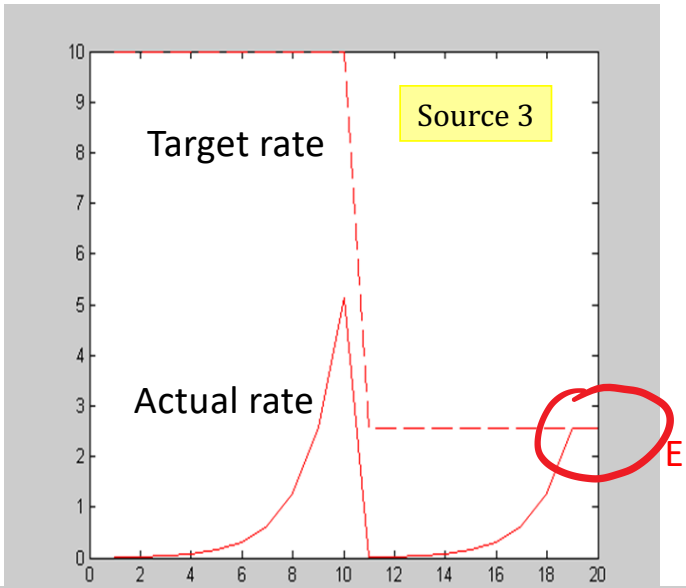


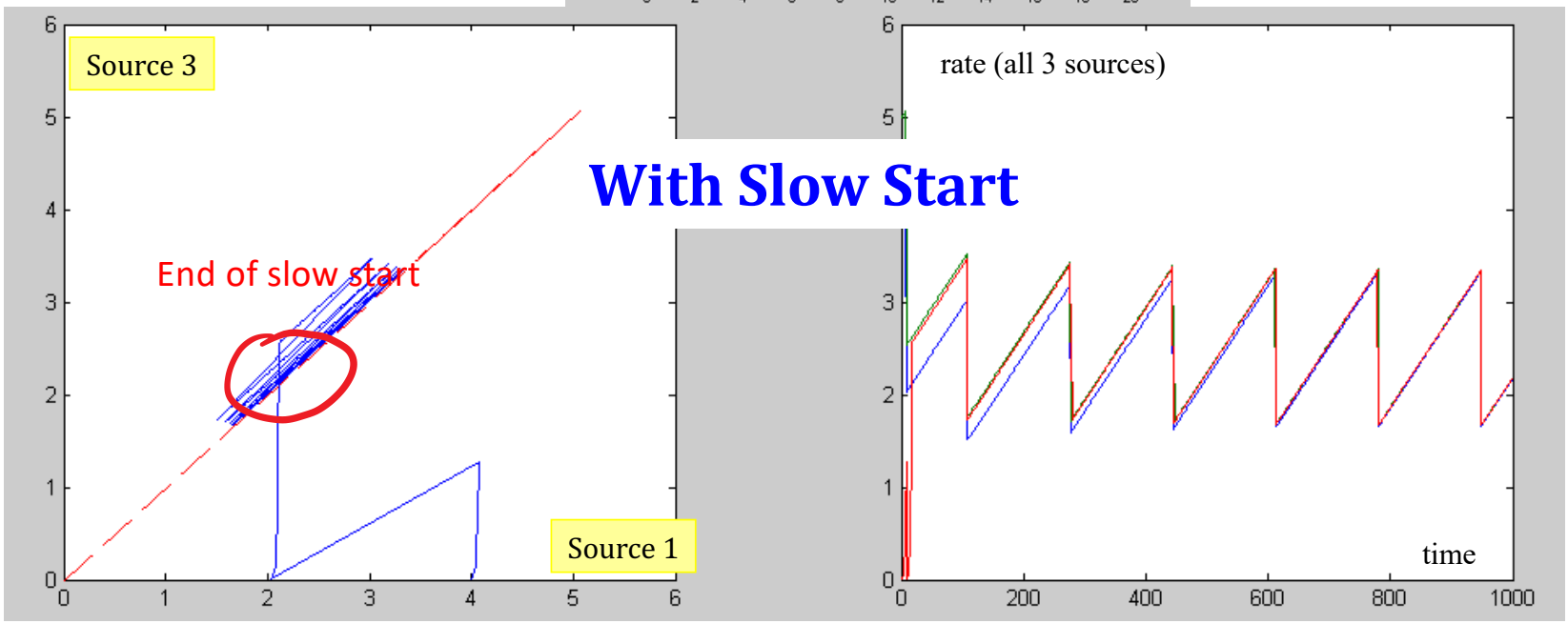
**Algorithm 2** Slow Start with the following parameters: AIMD constants multiplicative increase factor $w_0 > 1$; maximum rate $r_{\max} > 0$.

```
1:  rate ← v_0
2:  targetRate ← r_max
3:  do forever
4:  receive feedback
5:  if feedback is positive then
6:      rate ← w_0 · rate
7:      if rate ≥ targetRate then
8:          rate ← targetRate
9:          exit do loop
10:     end if
11: else
12:     targetRate ← max(u_1 · rate, v_0)
13:     rate ← v_0
14: end if
15: end do
```

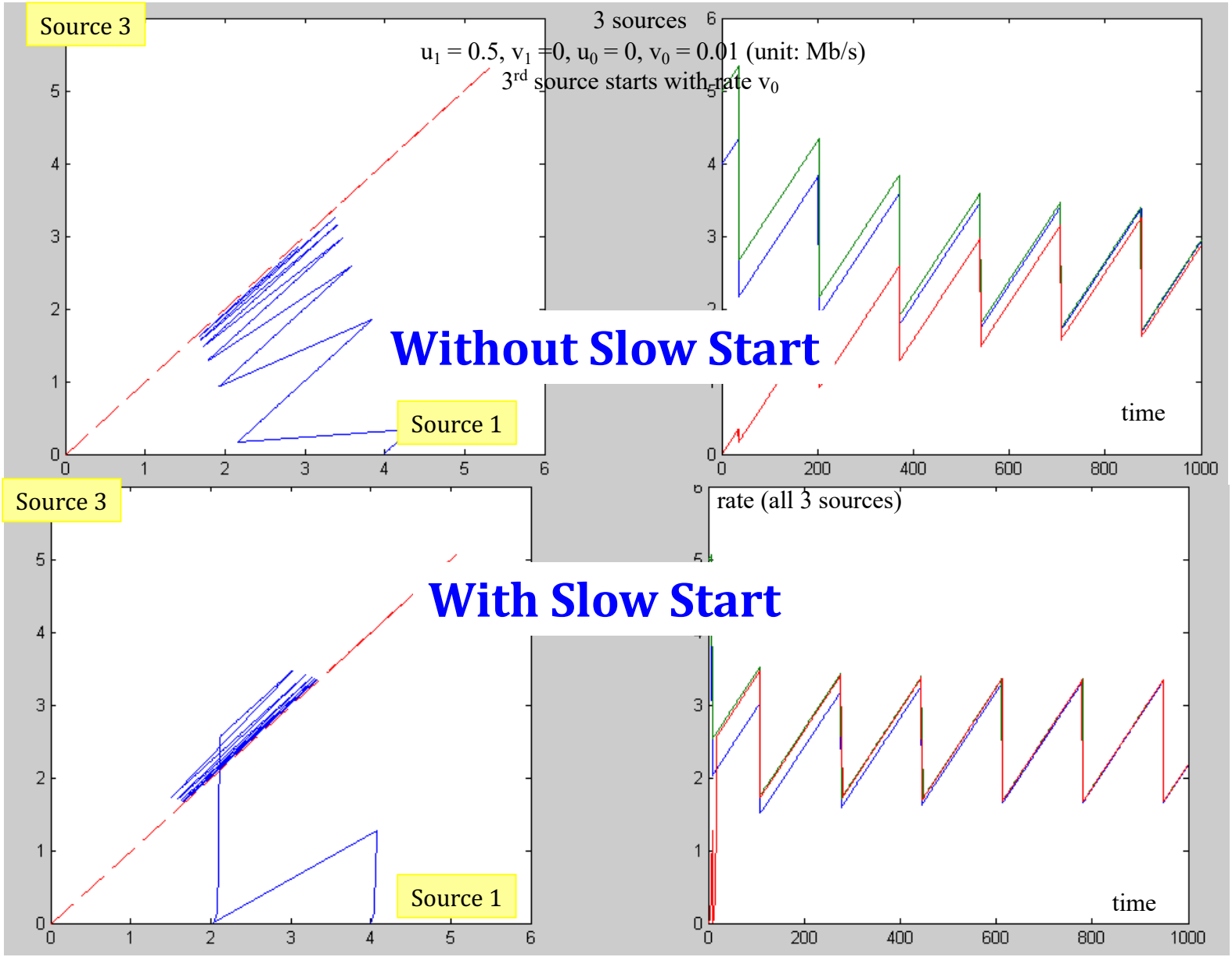
$w_0 = 2$

$u_1 = \frac{1}{2}$




**With Slow Start**


Source 3
3 sources
$u_1 = 0.5$, $v_1 = 0$, $u_0 = 0$, $v_0 = 0.01$ (unit: Mb/s)
3rd source starts with rate $v_0$
Without Slow Start
Source 1
time
Source 3
rate (all 3 sources)
With Slow Start
Source 1
time

# Summary

Congestion control is necessary to avoid inefficiencies and collapses

A congestion control scheme aims at allocating rates according to some form of fairness

In the internet, we use end-to-end congestion control with

- ≈ AIMD
- Slow Start
- and other refinements – see part 2.